Prémices d'une analyse syntaxique par transition pour des structures de dépendances non-projectives

Boris Karlov, Ophélie Lacroix

Prémices d'une analyse syntaxique par transition pour des structures de dépendance non-projectives

Boris Karlov[1], Ophélie Lacroix[2]
(1) Université de Tver, 33, rue Zheliabov, 170000, Tver, Russie
(2) LINA, 2, rue de la Houssinière 44322 Nantes Cedex 3
bnkarlov@gmail.com
ophelie.lacroix@univ-nantes.fr

RÉSUMÉ

L'article présente une extension de l'analyseur traditionnel en dépendances par transitions adapté aux dépendances discontinues et les premiers résultats de son entraînement sur un corpus de structures de dépendances de phrases en français. Les résultats des premières expérimentations vont servir de base pour le choix des traits des configurations de calcul bien adaptés aux dépendances discontinues pour améliorer l'apprentissage des dépendances tête.

ABSTRACT

Beginnings of a Transition-Based Parsing for Non-Projectives Dependency Structures

This paper presents an extension of the traditional transition-based dependency parser adapted to discontinuous dependencies and the first results of its training on a dependency tree corpus of French. The first experimental results will be useful for the choice of parsing configuration features well adapted to discontinuous dependencies in order to ameliorate learning of head dependencies.

MOTS-CLÉS : analyse syntaxique par transitions, structure de dépendance non-projective, grammaire catégorielle de dépendance.

KEYWORDS: transition-based parsing, non-projective dependency structure, dependency categorial grammar.

1 Introduction

Il existe différentes méthodes d'analyses syntaxiques permettant de traiter les phrases du français. Ces analyses peuvent être syntagmatiques (par constituants) (Kow *et al.*, 2006; Vanrullen *et al.*, 2006) ou en dépendance (Nasr, 2004; Brunet-Manquat, 2005; Alfared *et al.*, 2011). Elles peuvent être guidées par les règles d'une grammaire (probabiliste ou non) ou être entainées sur un corpus. Depuis plusieurs années, les différentes méthodes d'analyses en dépendance (voir (Kübler *et al.*, 2009)) gagnent en interêt dans le domaine de l'analyse syntaxique. Dans cet article nous nous plaçons dans le cas d'une analyse syntaxique en dépendance, par transition, entraînée sur un corpus correct par rapport à une grammaire de dépendance (Alfared *et al.*, 2011).
Les méthodes d'analyses en dépendance permettent de produire des représentations d'une phrase plus expressives sémantiquement que celles par constituant. D'autres formalismes comme les grammaires d'arbres adjoints sont parfois utilisés pour prendre en compte l'aspect sémantique

des phrases, mais ne permettent pas, par exemple, de révéler la relation de coréférence (Candito et Kahane, 1998). La représentation des phrases en structure de dépendance que nous avons choisi de mettre en avant ici, nous permet d'exprimer certaines relations qui ne sont pas toutes considérées dans les méthodes classiques par constituants. La figure 1 présente, entre autres, la relation distante existante entre les mots "*moins*" et "*que*" que l'on peut trouver lors d'une comparaison.

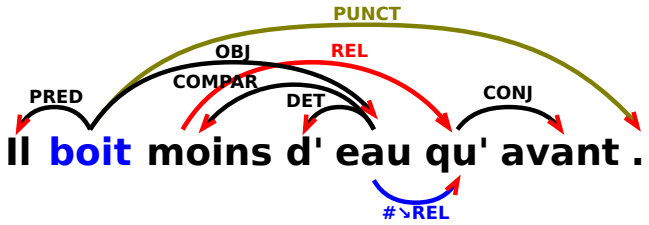

FIGURE 1 – Structure de dépendance de la phrase "Il boit moins d'eau qu'avant."

Il est donc possible de définir des dépendances croisées. Cependant une telle représentation (dite non-projective, voir la section 2.1) est peu utilisée dans le domaine de l'analyse en dépendance pour les phrases en français. Les analyses en dépendances courantes se basent principalement sur le même modèle que les analyses par constituants pour leur lien plus évident avec les grammaires formelles. Certains travaux consistent d'ailleurs à convertir des corpus annotés par constituants en corpus de dépendance (Candito *et al.*, 2009). Ce genre d'analyse produit donc des structures de dépendances projectives (sans dépendances croisées), proches des arbres syntagmatiques. On parle alors dans ce cas d'arbre de dépendance. Néanmoins, tout le potentiel des structures de dépendances n'est pas exploité lors d'une telle conversion. Comme le souligne (Rambow, 2010), une perte d'information serait inévitablement constatée si on tentait de convertir une structure de dépendance en structure par syntagme.
D'autre part, d'après (Nivre, 2011), les analyses du type analyse tabulaire (basée sur les premiers algorithmes tel que CKY ou Earley, voir (Kübler *et al.*, 2009)) ou analyse par satisfaction de contraintes (Maruyama, 1990) ne sont pas bien adaptées aux structures de dépendance non-projectives que nous souhaitons traiter. Le problème devient d'une complexité trop grande ou NP-complet. Les analyses par transition sur des structures de dépendance projectives pour des phrases en anglais donnent de bons résultats et l'adaptation aux structures de dépendance non-projectives se fait en une complexité au pire quadratique. Récemment, (Choi et Palmer, 2011) effectuent une analyse par transition sur des dépendances non-projectives pour des phrases en anglais. (Alfared *et al.*, 2011) travaillent sur un analyseur syntaxique semi-automatique permettant de produire des structures de dépendances projectives et non-projectives sur des phrases en français. Nous nous appuierons ici sur la grammaire catégorielle de dépendance qu'ils utilisent, (Dikovsky, 2011), pour définir les dépendances croisées existantes dans les structures de dépendance non-projectives. Nous allons alors présenter l'analyseur syntaxique par transition, que nous avons développé, permettant de définir des structures de dépendance non nécessairement projectives pour des phrases en français.

2 Structures de dépendance

Une structure de dépendance représente le résultat d'une analyse en dépendance pour une phrase donnée. Ces structures permettent de mettre en évidence les relations binaires entre les mots d'une phrase telles que les relations sujet-verbe, verbe-objet, etc. Les structures de dépendance sont à différencier des structures syntagmatiques qui sont le résultat d'une analyse par constituants. Mel'cuk met en évidence les différences, les avantages et les inconvénients de ces deux méthodes d'analyses dans (Mel'cuk, 1988) et propose la théorie "Sens-Texte" qui révèle l'aspect sémantique des dépendances entre les mots. Il met en avant, en outre, le fait que les structures de dépendances profondes sont invariantes selon l'ordre des mots dans la phrase. Ces idées sont aussi reprises plus récemment par (Kahane, 2001). Ici nous travaillons sur l'analyse des structures de dépendance de surface qui ont la particularité de garantir l'ordre des mots dans la phrase.

2.1 Projectivité, non-projectivité

De manière théorique, une structure de dépendance est un graphe orienté dont les noeuds sont les mots de la phrase à analyser et les arcs représentent les dépendances reliant ces mots. Dans une structure de dépendance, la relation de dominance est importante. Un mot domine de manière directe ses subordonnés, mais il domine aussi les subordonnés de ses subordonnés de manière transitive. Ainsi dans toute phrase la racine domine tous les mots. Un mot dominant est appelé un gouverneur.
Une structure de dépendance peut être projective ou non. Une structure est projective si pour chaque mot *w* tous les mots qui se trouvent entre *w* et ses subordonnés sont aussi dominés par *w*, comme illustré par la figure 2. Par ailleurs, les figures 1 et 3 présentent des structures de dépendances non-projectives. En effet, dans le cas de la figure 3, les mots "*attend*", "*son*" et "*mari*" sont compris entre le mot "*Elle*" et son subordonné "*pressée*" mais ne sont pas des subordonnés de "*Elle*", la structure n'est donc pas projective.

FIGURE 2 – Structure de dépendance de la phrase "La nature fait bien les choses."

2.2 Représentation utilisée

Les arcs représentant les dépendances sont de trois sortes différentes : projectifs, discontinus ou ancres. Les dépendances projectives sont des dépendances locales (courtes) qui ne se croisent pas les unes avec les autres. Les dépendances discontinues sont les dépendances non-projectives distantes qui peuvent croiser les dépendances projectives. Et les ancres sont des dépendances locales permettant de lier localement une dépendance discontinue. Effectivement, les subordonnés

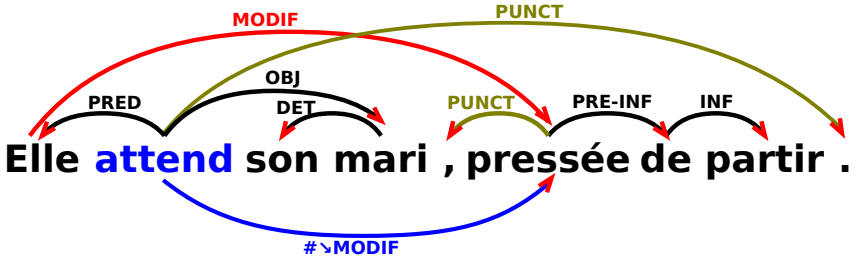

FIGURE 3 – Structure de dépendance de la phrase "Elle attend son mari, pressée de partir."

des dépendances ancres sont aussi subordonnés via des dépendances discontinues. Ces ancres sont importantes pour définir la grammaire présentée ensuite. La figure 3 permet d'illustrer chaque sorte d'arc énoncé ci-dessus. Dans cette figure, la dépendance discontinue est de type *modificateur* (le participe passé "*pressée*" est bien rattaché au pronom personnel "*Elle*"). Les dépendances discontinues, comme $MODIF$ [1] sont indiquées au dessus de la phrase tandis que les dépendances ancres liées aux dépendances discontinues, comme $\swarrow MODIF$, sont indiquées au dessous de la phrase. La *négation*, la *réfléxivité*, *l'apposition* et l'*aggrégation*, parmi d'autres types de dépendances, peuvent aussi engendrer des dépendances discontinues.

2.3 Grammaire catégorielle de dépendance (CDG)

(Dekhtyar et Dikovsky, 2008) exposent une méthode pour représenter les dépendances en catégories comme dans les grammaires catégorielles classiques (Bar-Hillel *et al.*, 1964). Cette grammaire a permis à (Alfared *et al.*, 2011) de produire le corpus que nous utiliserons ensuite pour notre analyse (voir section 4.1). L'idée est de représenter les dépendances projectives et les dépendances ancres par des catégories qui sont les types de ces dépendances et les dépendances non-projectives par des valences polarisées. Sur une phrase comme celle donnée en exemple dans la figure 4, on aura pour chaque mot les catégories suivantes : il $\mapsto [pred]$, y $\mapsto [\# \swarrow clit]^{\swarrow clit}$, est $\mapsto [\# \swarrow clit \backslash pred \backslash S / punct / aux]$, allé $\mapsto [aux]^{\nwarrow clit}$, . $\mapsto [punct]$.

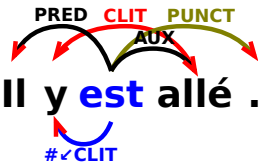

FIGURE 4 – Structure de dépendance de la phrase "Il y est allé."

Ces catégories sont ensuite utilisées dans une grammaire catégorielle enrichie de règles de dérivations, exposées dans la table 1, permettant de traiter les valences polarisées. La règle

1. $MODIF$ est le type attribué à une dépendance entre un mot et son modificateur, qui peut être un participe ou un adjectif. Les types de dépendances utilisés dans ces structures sont ceux employés par (Alfared *et al.*, 2011) pour construire leur corpus. Par ailleurs, les dépendances utilisées dans les figures de cet article sont étiquetées par les noms des groupes de dépendances auxquelles elles appartiennent (voir section 3.4) pour une illustration plus simple.

L permet d'éliminer les catégories comme dans les grammaires catégorielles classiques, et de concaténer les valences polarisées en une chaîne que l'on appelle *potentiel*. Pour la règle **I**, les valences sont traitées de la même manière tandis que la dérivation s'effectue sur les catégories itérables. De même, la règle Ω permet d'éliminer une catégorie itérable en conservant le potentiel tel qu'il était. Puis l'élimination des valences dans la dérivation (règle **D**) se fait sur le principe **FA** (First Available). Tout d'abord, des valences duales sont des valences de même catégorie dont les polarités sont $\swarrow$ et $\nwarrow$, ou $\nearrow$ et $\searrow$. Le principe FA indique que les valences duales les plus proches dans un potentiel sont des paires. Alors dans un potentiel $P_1(\swarrow C)P(\nwarrow C)P_2$, si $(\swarrow C)$ et $(\nwarrow C)$ (valences duales) n'apparaissent pas dans P, $(\swarrow C)$ et $(\nwarrow C)$ satisfont le principe **FA**.

$\mathbf{L}^1$	$C^{P_1}\left[C\backslash\beta\right]^{P_2} \vdash \left[\beta\right]^{P_1P_2}$
$\mathbf{I}^1$	$C^{P_1}\left[C^*\backslash\beta\right]^{P_2} \vdash \left[C^*\backslash\beta\right]^{P_1P_2}$
Ω^1	$\left[C^*\backslash\beta\right]^{P} \vdash \left[\beta\right]^{P}$
$\mathbf{D}^1$	$\alpha^{P_1(\swarrow C)P(\nwarrow C)P_2} \vdash \alpha^{P_1PP_2}$, si $(\swarrow C)(\nwarrow C)$ satisfait le principe **FA**

TABLE 1 – Règles de la grammaire catégorielle de dépendance généralisée

Un exemple d'arbre de dérivation, utilisant les valences polarisées pour les dépendances discontinues, dérivé de la phrase "*Il y est allé.*" est présenté par la figure 5.

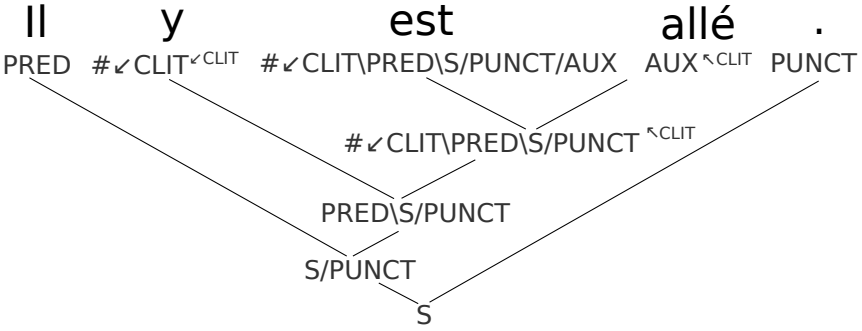

FIGURE 5 – Arbre de dérivation utilisant la CDG sur la phrase "Il y est allé."

3 Analyse par transition

Comme expliqué par (Nivre, 2008), une structure de dépendance peut être traduite en une suite de transitions. Le but d'un analyseur syntaxique par transition est de trouver une suite de transitions (opérations sur les configurations de calcul de l'analyseur) qui permette de construire une structure de dépendance correcte pour une phrase donnée. Pour faire ce travail, l'analyseur syntaxique doit s'appuyer sur un oracle qui lui indique quelle transition appliquer dans telle ou telle configuration. Les configurations et transitions sont des éléments essentiels de l'analyse par transition, qui peuvent varier selon l'utilisation que l'on veut en faire. Kübler, Donald et Nivre définissent clairement leurs emplois dans (Kübler *et al.*, 2009). Ici nous présentons un analyseur

par transition adapté aux dépendances discontinues, que nous avons développé. Le travail consiste tout d'abord à générer un oracle à partir d'un corpus de référence. Chaque structure de dépendance du corpus est traduite en un ensemble de couples configuration-transition qui sont ajoutées à l'oracle selon le nombre d'occurences d'une transition pour une configuration [2] donnée. L'oracle enregistre donc seulement la meilleure transition possible pour une configuration. Cet oracle sera alors utilisé pour déterminer une suite de transitions à appliquer, pour chaque phrase à analyser, permettant de construire des structures de dépendance. L'analyseur est donc déterministe car il s'appuie sur l'oracle pour choisir la transition à appliquer à chaque étape de l'analyse.

3.1 Configurations

Une configuration, pour une phrase, est un "état" dans lequel est l'analyseur syntaxique analysant cette phrase. On peut y appliquer une transition qui permettra de passer dans la configuration suivante.
Soit une phrase $s = w_1 w_2 ... w_n$. Une configuration pour une phrase est un quintuplet $(\sigma, \beta, \theta, i, E)$ où σ est une pile de mots w_i dans s, β est un tampon de mots w_i dans s, θ est un tampon de valences (un potentiel), i est la position du mot courant, et E est un ensemble d'arcs (w_i, r, t, w_j) où r est l'étiquette de l'arc (le type de la dépendance) et t est la sorte de l'arc (projectif, discontinu, ancre). La configuration initiale c_0 pour chaque phrase est :

$$([], [w_1, ..., w_n], [], 1, \emptyset).$$

La configuration finale, pour n'importe quel σ et E, est :

$$(\sigma, [], [], n+1, E).$$

3.2 Transitions

Les transitions sont des opérations s'appliquant aux configurations pour obtenir les configurations suivantes. Celles-ci sont présentées dans la table 2. Ces transitions se basent sur les transitions évoquées par (Kübler *et al.*, 2009). Nous y ajoutons des éléments permettant de traiter les structures de dépendances projectives aussi bien que les structures non-projectives. En effet, la transition *PutPotential* ajoute les valences polarisées dans le potentiel θ selon les règles de la table 1 de la grammaire catégorielle de dépendance vu à la section 2.3. Puis les transitions *DistLeft* et *DistRight* permettent de définir les dépendances discontinues de la structure de dépendance en éliminant les valences duales du potentiel θ, suivant la règle **D**. En outre, les transitions *LocalLeft* et *LocalRight* servent à définir les dépendances projectives ainsi que les dépendances ancres. Les ancres permettent de détecter les subordonnés discontinus.
Pour convertir une structure de dépendance en une suite de transitions, on utilise un algorithme qui applique les transitions dans un certain ordre. Pour cela, on considère toujours le mot en haut de la pile σ comme le mot courant de la configuration. L'algorithme essaie d'abord de trouver les dépendances discontinues de la structure. Donc dans un premier temps, il tente d'ajouter une valence dans le potentiel θ en appliquant la transition *PutPotential* si le mot courant est impliqué

2. Une configuration est représentée par un vecteur de traits : image de taille réduite de la configuration, voir section 3.4

dans une dépendance discontinue. Puis on applique les transitions *DistLeft* et *DistRight*, pour définir les dépendances distantes trouvées, jusqu'à ce qu'il n'y ait plus de valences duales. Ensuite l'algorithme traite les dépendances locales. Il commence donc par appliquer toutes les transitions *LocalLeft* possible pour le mot courant, de son subordonné le plus proche au plus à gauche. Puis si ce mot a des dépendances à droite, on le garde sur la pile et on applique la transition *Put* pour empiler le mot suivant sur σ. Néanmoins, si le mot courant n'a pas de subordonné droit et a un gouverneur sur la gauche alors il est possible d'appliquer la transition *LocalRight*. Le dernier cas possible est si le mot courant n'a plus aucun subordonné sur la droite et n'a pas non plus de gouverneur. Alors il s'agit de la racine et le processus est terminé.

Transition	Effet de son application sur une configuration
PutPotential $(p_1, ..., p_m)$	$(\sigma, w_k \mid \beta, \theta, k, E) \Rightarrow (\sigma, w_k \mid \beta, \theta p_1^k ... p_m^k, k+1, E)$ où les p_j sont des valences.
Put	$(\sigma, w_i \mid \beta, \theta, k, E) \Rightarrow (\sigma \mid w_i, \beta, \theta, next(k,i), E)$ où $next(k,l) = \begin{cases} k+1 & si\ k = l \\ k & sinon \end{cases}$
LocalLeft(d)	$(\sigma \mid w_i, w_j \mid \beta, \theta, k, E) \Rightarrow (\sigma, w_j \mid \beta, \theta, k, E \cup \{(w_i, d, TYPE(d), w_j)\})$
LocalRight(d)	$(\sigma \mid w_i, w_j \mid \beta, \theta, k, E) \Rightarrow (\sigma, w_i \mid \beta, \theta, next(k,j), E \cup \{(w_i, d, TYPE(d), w_j)\})$
DistLeft(v)	$(\sigma, \beta, \theta_1 \swarrow v^i \theta_2 \nwarrow v^j \theta_3, k, E) \Rightarrow (\sigma, \beta, \theta, k, E \cup \{(w_j, v, discontinuous, w_i)\})$ si $\swarrow v^i \nwarrow v^j$ est la paire la plus à gauche satisfaisant la condition FA.
DistRight(v)	$(\sigma, \beta, \theta_1 \nearrow v^i \theta_2 \searrow v^j \theta_3, k, E) \Rightarrow (\sigma, \beta, \theta, k, E \cup \{(w_i, v, discontinuous, w_j)\})$ si $\nearrow v^i \searrow v^j$ est la paire la plus à gauche satisfaisant la condition FA.

TABLE 2 – Les transitions utilisées par l'analyseur

3.3 Exemple de conversion

Pour illustrer la méthode de conversion d'une structure de dépendance en séquence de transitions voici un exemple pour la phrase "*Cette victoire, elle l'a méritée.*" dont la structure de dépendance est présentée par la figure 6. Cette phrase comprend des dépendances discontinues de type *clitique* et *coréférence*. La suite de transitions appliquées à cette phrase et les éléments de chaque configuration en découlants sont illustrés dans la table 3.

Transition	Pile σ	Tampon β	Potentiel θ	Ensemble d'arcs E
	[]	[Cette, victoire , elle l' a méritée .]	[]	$\varnothing$
Put	[Cette]	[victoire , elle l' a méritée .]	[]	$\varnothing$
PutPotential	[Cette]	[victoire , elle l' a méritée .]	[↙coref]	$\varnothing$
LocalLeft	[]	[victoire , elle l' a méritée .]	[↙coref]	E = (victoire, projective, det, Cette)
Put	[victoire]	[, elle l' a méritée .]	[↙coref]	E
LocalRight	[]	[victoire elle l' a méritée .]	[↙coref]	E = E ∪ (victoire, projective, punct, ,)
Put	[victoire]	[elle l' a méritée .]	[↙coref]	E
Put	[victoire elle]	[l' a méritée .]	[↙coref]	E
PutPotential	[victoire elle]	[l' a méritée .]	[↙coref↖coref]	E
PutPotential	[victoire elle]	[l' a méritée .]	[↙coref↖coref↙clit]	E
DistLeft	[victoire elle]	[l' a méritée .]	[↙clit]	E = E ∪ (l', discontinuous, coref, victoire)
Put	[victoire elle l']	[a méritée .]	[↙clit]	E
LocalLeft	[victoire elle]	[a méritée .]	[↙clit]	E = E ∪ (a, anchor, clit, l')
LocalLeft	[victoire]	[a méritée .]	[↙clit]	E = E ∪ (a, projective, pred, elle)
LocalLeft	[]	[a méritée .]	[↙clit]	E = E ∪ (a, anchor, coref, victoire)
Put	[a]	[méritée .]	[↙clit]	E
PutPotential	[a]	[méritée .]	[↙clit↖clit]	E
DistLeft	[a]	[méritée .]	[]	E = E ∪ (méritée, discontinuous, clit, l')
LocalRight	[]	[a .]	[]	E = E ∪ (a, projective, aux, méritée)
Put	[a]	[.]	[]	E
LocalRight	[]	[a]	[]	E = E ∪ (a, projective, punct, .)

TABLE 3 – Conversion de la structure de dépendance de la phrase "Cette victoire, elle l'a méritée." en une suite de transitions

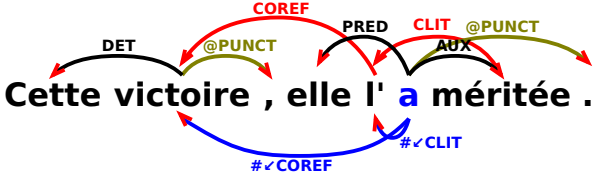

FIGURE 6 – Structure de dépendance de la phrase "Cette victoire, elle l'a méritée."

3.4 Oracles

L'idée de l'analyse par transition est de représenter partiellement les éléments des configurations (dont la taille n'est pas limitée) par les valeurs de vecteurs de traits. Le problème du choix des traits se pose alors. Différents traits peuvent être choisis pour entraîner un oracle à partir d'un corpus de structure de dépendance. Nous expérimentons ici avec huit vecteurs de traits différents. On peut donc choisir :
– l'utilisation ou non du nom complet de la classe grammaticale affectée au mot dans les structures de dépendance
– l'utilisation ou non des groupes de dépendances à la place des noms exacts des dépendances
– la taille du vecteur de traits.

Pour la classe grammaticale, on peut choisir d'utiliser le nom simple (court) de la classe ou le nom enrichi d'un paramètre (long). Par exemple pour le cas d'un mot de la classe adjectif quantificateur, le nom court correspondra à *Adj* et le nom long à *Adjquantifier*. La classe courte *Adj* rassemble les adjectifs modificateurs, quantifiants, restrictifs, comparatifs, etc... Un autre exemple est celui de la classe des noms (*N*). Elles peut regrouper les classes de noms longs *Nproper* (noms propres), *Ncommon* (noms communs), *Ntime* (noms à référence temporelle), etc...

Par ailleurs, les dépendances avec des fonctions syntaxiques proches peuvent être regroupées dans des groupes de dépendances. Il est alors possible de sélectionner une option traitant les dépendances selon le groupe de dépendances auxquelles elles appartiennent ou selon leurs types de dépendances précis. Par exemple pour le cas d'un mot dont la dépendance est du type objet accusatif, on utilisera par défaut le type de dépendance complet *a-obj*, mais on emploiera le type *OBJ* pour une représentation des types par groupe. Dans ce cas là, le type *OBJ* pourra aussi être assigné à une dépendance de type *g-obj* (objet génitif), *l-obj* (objet locatif) ou *o-obj* (objet oblique).

Le vecteur de trait peut être de taille 6 ou 8. Dans les deux cas, les vecteurs de traits, pour une configuration donnée à un moment de l'analyse, sont constitués de :
– la classe grammaticale du mot au sommet de la pile σ
– la classe grammaticale du premier mot du tampon β de mots
– le type de la dépendance la plus à gauche (c'est-à-dire, dont le subordonné est le plus à gauche) du mot en haut de la pile σ suivi de la sorte de l'arc (discontinu, projectif, ancre)
– le type de la dépendance la plus à droite du mot en haut de la pile σ suivi de la sorte de l'arc
– la dernière valence du potentiel et son orientation
– l'avant-dernière valence du potentiel et son orientation

Si le vecteur est de taille 8, il possède les deux traits suivants en plus :
– le type de la dépendance la plus à gauche du premier mot du tampon β suivi de la sorte de l'arc
– le type de la dépendance la plus à droite du premier mot du tampon β suivi de la sorte de l'arc

Ainsi nous générons huit oracles différents, exposés dans la table 4. L'oracle 1 est ainsi le plus général et l'oracle 8 est le plus précis.

	Taille du vecteur		*Classe grammaticale*		*Groupe*	
	6	8	*courte*	*longue*	*oui*	*non*
Oracle 1	√		√		√	
Oracle 2	√		√			√
Oracle 3	√			√	√	
Oracle 4	√			√		√
Oracle 5		√	√		√	
Oracle 6		√	√			√
Oracle 7		√		√	√	
Oracle 8		√		√		√

TABLE 4 – Les paramètres des différents oracles

4 Résultats des analyses et discussion

4.1 Corpus de dépendance

Le corpus utilisé pour cette analyse par transition est constitué d'un ensemble de corpus de phrases de style grammatical différent. Il a été annoté semi-automatiquement par des membres de l'équipe TALN du Lina et leurs associés, en utilisant le système *CDG Lab* (Alfared *et al.*, 2011) et une grammaire catégorielle du français (Dikovsky, 2011). Lors de chaque analyse syntaxique 90% du corpus est employé comme corpus d'entraînement pour produire un oracle et les 10% restants sont exploités par l'analyse syntaxique par transition et traduits en structures de dépendances. Le choix des phrases utilisées pour l'apprentissage et pour l'analyse se fait aléatoirement. Le corpus complet (apprentissage et analyse) comprend 2557 structures de dépendances dont 33490 mots. 41% des structures de dépendances de ce corpus contiennent au moins une dépendance discontinue mais les dépendances discontinues représentent seulement 4% du nombre de dépendances total du corpus.

4.2 Méthode d'expérimentation

Les fichiers de sortie de l'analyseur sont similaires aux fichiers d'entrée, de manière à ce que l'on puisse retrouver pour chaque phrase, les dépendances correctements assignées ou non entre les unités lexicales. Pour estimer les résultats, on calculera la f-mesure sur les têtes des dépendances. C'est à dire que l'on se contentera de vérifier si chaque mot de la phrase reçoit bien la dépendance du bon type. Toutefois, les dépendances discontinues ne sont pas comptabilisées, car les mots recevant une telle dépendance reçoivent aussi une dépendance ancre du même type. Le choix de calculer les résultats uniquement sur les têtes des dépendances plutôt que sur les dépendances elles-mêmes dépend de deux aspects importants :

– le volume du corpus n'est pas suffisant pour l'instant pour espérer avoir les bons assignements des dépendances et de leurs types en même temps
– ces types de résultats s'accordent avec des travaux futurs qui consisteront à déterminer les bonnes têtes des dépendances avant d'effectuer une analyse syntaxique en dépendance à partir de la CDG.

Le principe est donc de collecter, pour chaque type de dépendance apparaîssant dans le corpus d'entrée ou de sortie, le nombre de fois où celui-ci est assigné dans le corpus d'entrée, le nombre de fois où celui-ci est attribué à un mot du corpus de sortie et le nombre de fois où celui-ci est attribué au bon mot dans le corpus de sortie. La table 5 donne les résultats de ces assignations sur une analyse d'un corpus d'analyse possible (10% du corpus total, voir section 4.1) pour certains types de dépendances. Par exemple, le type *S* (la tête de la phrase) a été assigné sur 38 mots, dont 37 fois correctement, parmi les 255 mots typés *S* dans le corpus. Le type *DET* a toujours été assigné correctement sur les 117 fois où il a été trouvé pour un mot.

Type des dépendance	Nb d'occurences corpus d'entrée	Nb d'occurences corpus de sortie	Nb d'attributions correctes
S	38	255	37
PRED	169	314	168
DET	117	331	117
OBJ	69	218	64
COPUL	22	79	21
MODIF	26	113	24
CLIT	48	124	39
PUNCT	116	404	109

TABLE 5 – Exemple de résultats d'assignements des types (pour quelques types donnés) par comparaison entre un corpus d'entrée et de sortie

Pour chaque type de dépendance i de la table ainsi établie, on calcule la présicion et le rappel de la manière suivante :

$$\text{précision}_i = \frac{\text{Nb d'attributions correctes pour i}}{\text{Nb d'occurences corpus de sortie pour i}} \qquad \text{rappel}_i = \frac{\text{Nb d'attributions correctes pour i}}{\text{Nb d'occurences corpus d'entrée pour i}}$$

4.3 Résultats des expérimentations

Les valeurs de la f-mesure calculées selon la méthode décrite à la section 4.2 sont indiquées dans la table 6 en fonction de chaque oracle (voir la table 4 pour les caractéristiques des oracles). Les résultats sont une moyenne mesurée sur 20 itérations comprenant chacune la séparation du corpus en un corpus d'entraînement et un corpus de test, l'entraînement de l'oracle, l'analyse syntaxique par transition et le calcul de la f-mesure.

L'analyse des résultats montre que le choix du vecteur de traits pour l'oracle influe de manière significative sur l'attribution correcte ou non des types de dépendances sur les mots. Les meilleurs résultats se font avec les oracles dont les vecteurs de traits sont les plus informatifs. Le choix d'une classe grammaticale longue et d'une taille de vecteur de 8 est préférable pour obtenir de meilleurs résultats. En outre, le choix d'utiliser les groupes à la place des types exacts semble dépendre de la longueur de la classe grammaticale. Ces deux paramètres ont en effet un lien. Par exemple, si un mot de classe grammaticale N[3] à reçu le type de dépendance *a-obj*, en utilisant

3. voir section 3.4

	Oracle 1	Oracle 2	Oracle 3	Oracle 4
Précision	0.373	0.282	0.567	0.615
Rappel	0.187	0.151	0.258	0.294
F-mesure	0.250	0.197	0.355	0.398
	Oracle 5	Oracle 6	Oracle 7	Oracle 8
Précision	0.463	0.393	0.635	0.697
Rappel	0.192	0.169	0.313	0.362
F-mesure	0.272	0.237	0.420	0.477

TABLE 6 – Résultats des expérimentations sur les sorties de l'analyse selon les différents oracles

l'oracle 6, alors qu'il est de type *a-obj-d*, la dépendance sera considérée comme mal attribuée. Alors qu'avec l'oracle 5, la dépendance sera seulement *OBJ* dans les deux cas et sera donc correcte pour ce mot. Inversement, le problème se pose d'une autre manière entre les oracles 7 et 8. Lors de l'entraînement des oracles, la transition choisie dépend de la fréquence d'apparition d'une configuration dans le corpus. Un mot de classe grammaticale *Ntime* peut recevoir une dépendance de type objet ou préposition par exemple. Il se pourrait alors qu'en utilisant les groupes, l'ensemble des dépendances de type *PREPOS* soient plus nombreuses que l'ensemble des dépendances de types *OBJ* alors que sans utiliser les groupes, les dépendances de type *a-obj* soient plus nombreuses que chacune des dépendances de type *prepos-x* (où x=g|d|l|o|A|sel). Donc dans un cas, l'oracle assignerait une dépendance de type préposition alors que dans l'autre, il assignerait une dépendance de type objet.

En outre, la taille du vecteur de trait est significative dans le sens où les options ajoutées au vecteur de taille 8 apportent une information importante lors de l'analyse. En effet, de cette manière on connait la dépendance la plus à gauche et la dépendance la plus à droite de chacun des deux mots traités (le mot en haut de la pile σ et le premier mot du tampon β). On peut savoir par exemple si un verbe, étant la tête de la phrase, a déjà un sujet ou non. Alors qu'avec un vecteur de trait de taille 6, cette information n'est pas conservée et plusieurs sujets peuvent être attribués dans une même phrase lorsqu'il n'y a pas lieu de le faire.

4.4 Perspectives et améliorations à venir

Le choix d'une transition lors de l'analyse se fait de manière déterministe car, dans l'oracle, il n'y a qu'une seule transition possible pour une configuration donnée. En outre, plus les options choisies pour l'oracle sont généralistes, moins il y a de choix car des configurations différentes se retrouvent confondues dans un même vecteur de traits qui dirige l'analyse vers une seule transition pour des cas très différents. Il sera donc nécessaire par la suite de mémoriser dans l'oracle, en plus de l'ensemble configuration-transition le meilleur, quelques autres possibilitées de transition pour la configuration considérée.

De plus, l'analyse étant déterministe, celle-ci s'interrompt lorsqu'il n'y a pas de transition applicable. Beaucoup de structures de dépendances résultantes sont alors incomplètes ou parfois complétement vides. La proposition précedente, permettrait donc ensuite de pouvoir faire un

retour en arrière dans la séquence de transitions. Le nombre de retour en arrière possible pour une phrase devra être limité et le choix d'une nouvelle transition pourra se baser sur la seconde meilleure transition pour une certaine configuration. Ce problème est aussi une cause du faible score du rappel dans toutes les analyses. En effet, beaucoup de mots n'ont tout simplement pas d'attribution de type alors que les mots ayant une dépendance sont plus souvent correctement typés, d'où une précision meilleure que le rappel (voir exemple table 5).
Le choix d'une première transition puis d'une potentielle seconde meilleure transition devra se baser sur des données probabilistes. Ainsi, pour établir ces données il sera nécessaire d'expérimenter différents lissages avant de mémoriser les ensembles configuration-transitions dans l'oracle.

D'après les résultats des expérimentations, les oracles utilisant un vecteur de taille 8 sont plus efficaces car ils mémorisent les types des dépendances gauches et droites des deux mots traités lors de l'analyse. Cependant, ces informations ne sont pas pertinentes dans tous les cas. Notamment, le fait de savoir qu'un mot a une dépendance de type modificateur n'est pas déterminant pour la suite car rien n'empêche ce mot d'avoir d'autres modificateurs. Tous les types itérables, comme les modificateurs, les circonstantiels, les coordinations verbales, assignés à une dépendance ne seront donc pas mémorisés. Nous garderons uniquement des informations appropriées telles que les types prédicat, déterminant, objet, etc...

5 Conclusion

Les objectifs de cet article étaient de proposer des résultats d'expérimentations avec un analyseur syntaxique par transitions étendu aux dépendances discontinues et de préparer la base d'une comparaison de ces résultats avec un oracle guidant l'analyse de la CDG du français.
L'analyse par transition décrite dans cet article ne permet pas encore d'obtenir des résultats suffisants pour obtenir une bonne analyse en dépendance du français. Cependant, le système de transitions mis en place pour correspondre avec la grammaire CDG est tout à fait adaptée à l'ajout des dépendances discontinues par valences polarisées. De plus, nous voulions mettre en évidence les intérêts et les problèmes de ces premières expérimentations sur des phrases en français pour mettre en place un plan d'améliorations de l'analyse par transitions. Dans l'état actuel, l'algorithme établi pour analyser les phrases en structures de dépendances est déterministe et ne permet pas de faire des choix bien adaptés aux configurations traitées. L'aspect probabilistes mis en place pour calculer la fréquence d'apparition des configurations dans l'analyse des phrases du corpus devra être exploité pour sélectionner les meilleures transitions possibles pour une configuration et pas seulement la meilleure. En outre, les vecteurs de traits devront être mieux adaptés à l'information dont l'analyseur a besoin pour faire les bons choix.
Dès lors, les prochains résultats permettront de comparer cette analyse avec l'oracle qui guide l'analyse de la CDG du français et ainsi d'établir de nouvelles méthodes d'amorçages pour attribuer les têtes des dépendances avant l'analyse de la CDG.

Références

ALFARED, R., BÉCHET, D. et DIKOVSKY, A. (2011). "CDG Lab" : a Toolbox for Dependency Grammars and Dependency Treebanks Development. *In DEPLING 2011 (International Conference on Dependency Linguistics)*, Barcelona.

BAR-HILLEL, Y., GAIFMAN, C. et SHAMIR, E. (1964). *On Categorial and Phrase Structure Grammars*, pages 99–115. Addison-Wesley.

BRUNET-MANQUAT, F. (2005). Improving dependency analysis by Syntactic parser combination. *In NLP-KE 2005 (Natural Language Processing and Knowledge Engineering)*, Wuhan.

CANDITO, M., CRABBÉ, B., DENIS, P. et GUÉRIN, F. (2009). Analyse syntaxique du français : des constituants aux dépendances. *In TALN 2009 (Traitement automatique des langues naturelles)*, Senlis.

CANDITO, M.-H. et KAHANE, S. (1998). Une grammaire TAG vue comme une grammaire Sens-Texte précompilée. *In TALN 1998 (Traitement automatique des langues naturelles)*, Paris.

CHOI, J. D. et PALMER, M. (2011). Getting the Most out of Transition-based Dependency Parsing. *In ACL 2011 (Association for Computational Linguistics)*, Portland.

DEKHTYAR, M. et DIKOVSKY, A. (2008). *Generalized Categorial Dependency Grammars*, pages 230–255. LNCS 4800. Springer.

DIKOVSKY, A. (2011). Categorial Dependency Grammars : from Theory to Large Scale Grammars. *In DEPLING 2011 (International Conference on Dependency Linguistics)*, Barcelona.

KAHANE, S. (2001). Grammaires de dépendance formelles et théorie Sens-Texte. *In TALN 2001 (Traitement automatique des langues naturelles)*, Tours.

KOW, E., PARMENTIER, Y. et GARDENT, C. (2006). SemTAG, the LORIA toolbox for TAG-based Parsing and Generation. *In TAG+8 (The Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms)*, Sydney.

KÜBLER, S., MCDONALD, R. et NIVRE, J. (2009). *Dependency parsing*. Morgan et Claypool.

MARUYAMA, H. (1990). Structural disambiguation with constraint propagation. *In ACL 1990 (Association for Computational Linguistics)*, Pittsburgh.

MEL'CUK, I. (1988). *Dependency syntax : Theory and Practice*. State University of New York Press.

NASR, A. (2004). *Analyse syntaxique probabiliste pour grammaires de dépendances extraites automatiquement*. Habilitation à diriger des recherches, Université Paris 7, UFR d'informatique.

NIVRE, J. (2008). *Algorithms for Deterministic Incremental Dependency Parsing*, pages 513–553. Volume 34 (4). Massachusetts Institute of Technology.

NIVRE, J. (2011). Bare-Bones Dependency Parsing. *In NODALIDA 2011 (Nordic Conference of Computational Linguistics)*, Riga.

RAMBOW, O. (2010). The Simple Truth about Dependency and Phrase Structure Representations. *In NAACL HTL 2010 (North American Chapter of the Association for Computational Linguistics - Human Language Technologies)*, Los Angeles.

VANRULLEN, T., BLACHE, P. et BALFOURIER, J.-M. (2006). Constraint-Based Parsing as an Efficient Solution : Results from the Parsing Evaluation Campaign EASy. *In LREC 2006 (Conference on Language Resources and Evaluation)*, Genoa.